Code Generation

```
sum(n, m){
  if( n == m ) return m;
  else return n + sum( n+1, m);
}

main(){
  print(sum(1,10));
}
```

```
(fun main
    (print
        (call sum 1 10 )))
(fun sum
    (else
        (== #1 #2 )
        (return #2 )
        (return (+ #1 1 )#2 )))))
```

```
:main
    fun.1
    lit.1
    lit.10
    lit.0
    call.sum2
    sys.1
    ret.1
:sum
    fun.1
    get.2
    get.1
    eq
    jf.L18
    get.1
    ret.3
    jmp.L26
:L18
    get.2
    get.2
    lit.1
    add
    get.1
    call.sum
    add
    ret.3
:L26
    ret.3
```

s-code
http://www.cp.eng.chula.ac.th/faculty/pjw/project/som/s-code.htm

fix 32-bit width

  arg:24   op:8

```
arith:   add, sub, mul, div, mod
logic:   band, bor, bxor, shl, shr, not,
         eq, ne, lt, le, gt, ge
data:    ld, st, ldx, stx, lit, get, put
control: call, ret, case, fun, jt, jf, jmp
other:   inc, dec, callt, sys
```

36 instructions